LIST OF CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application.

1. (currently amended) A data structure, comprising:

a root level having a node group, the node group having k number of nodes, each of the k number of nodes sharing a pointer, each of the k number of nodes stored contiguously in memory;

a second level having one supernode, the supernode having k number of node groups, each of the k number of node groups includes k nodes, the k nodes of the supernode being stored contiguously in memory;

a first level memory storing the root level node group; and

a second level memory storing the second level supernode, wherein the first level memory and the second level memory may be accessed simultaneously, wherein the pointer references the contiguously stored k nodes of the supernode and each of the first level memory and the second level memory are associated with corresponding comparator blocks, the comparator blocks configured to examine hole counters representing the number of holes in the supernode.

2. (previously presented) The data structure of claim 1, further comprising a hole as a k node, the hole representing an absent value.

3. (previously presented) The data structure of claim 1, wherein the k number of node groups are siblings of each other, such that only one sibling node is needed for any given path.

4. (currently amended) The data structure of claim 1, wherein the arrangement of the supernode allows for speculatively reading a child node ~~children~~ before an exact child node is known.

5. (original) The data structure of claim 4, wherein the determination of the exact desired child proceeds in parallel with the retrieval of the supernode.

6. (previously presented) The data structure of claim 1, further comprising:

a third level having k number of supernodes, each of the k number of supernodes of the third level having k number of node groups, each of the k number of node groups includes k nodes, wherein the k number of supernodes of the third level are referenced to the second level by k pointers associating each one of the k number of node groups of the second level to a corresponding one of the k number of supernodes of the third level.

7. (original) The data structure of claim 2, further comprising a remove or delete operation which does not require a last value to be moved into a root node.

8. (previously presented) The data structure of claim 7 wherein the remove or delete operation comprises:

removing a value from the root node; and

percolating the hole associated with the removal of the value from the root node down the data structure.

9. (previously presented) The data structure of claim 7, wherein the data structure contains a hole counter that counts a number of holes below the pointer, the hole counter being associated with the pointer, the hole counter representing the number of holes in the supernode below the pointer.

10. (previously presented) The data structure of claim 9, wherein the remove operation comprises incrementing the hole counter associated with the pointer when the pointer is traversed.

11. (original) The data structure of claim 2, further comprising an insert operation for percolating a value to be inserted starting at the root level and proceeding towards the bottom level.

12. (original) The data structure of claim 10, wherein an insert operation comprises:

percolating a value to be inserted starting at the root value;

in one or more pointers, each pointer being associated with a hole counter that tracks the number of available holes, percolating the add value down a node in which the hole counter contains a value greater than zero; and

decrementing the selected hole counter by one.

13. (original) The data structure of claim 1, wherein an insert operation and a remove operation access the data structure in a top-to-bottom order.

14. cancelled

15. cancelled